Quick Guide

How to create a Notebook with biosignalsnotebooks template

Creating a Notebook is a very instructive process, not only for the user but also to the creator while searching for the best approach to transmit knowledge.

Through the created explanations, figures and code, all **biosignalsplux** users can easily start processing the signals that they acquired, entering in the amazing world of digital signal analysis while exploring **biosignalsnotebooks** environment.

All Notebooks, inside **biosignalsnotebooks** environment, have a common style that needs to be respected for the current creations and also for the new Notebooks that will be created.

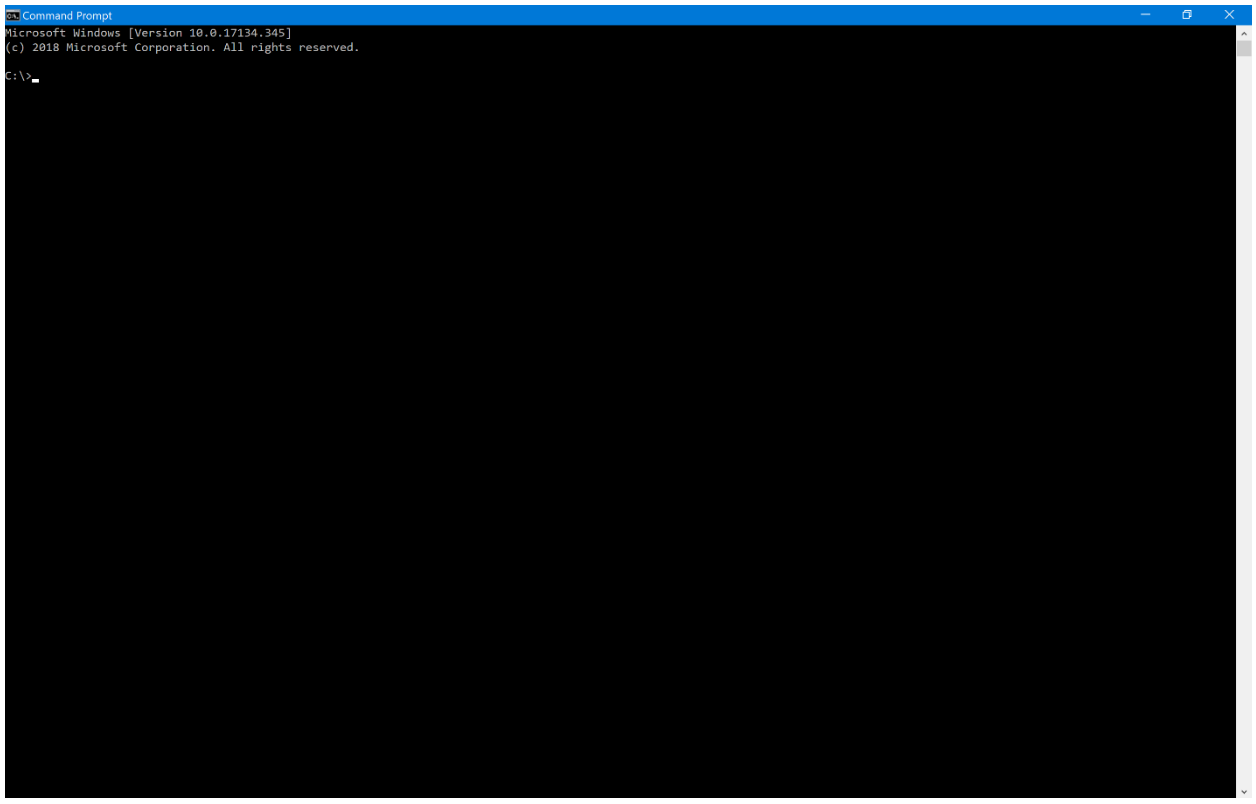
To simplify the procedure of creating a new Notebook, it is available a specialized module, called "factory", inside **biosignalsnotebooks** Python package.

This document will guide you with the steps description and some illustrative images.

☌ A - Creation of a biosignalsnotebooks project folder, which we proudly call "biosignalsnotebooks_environment"

A1 – Open a command window (type "cmd" at the start menu)

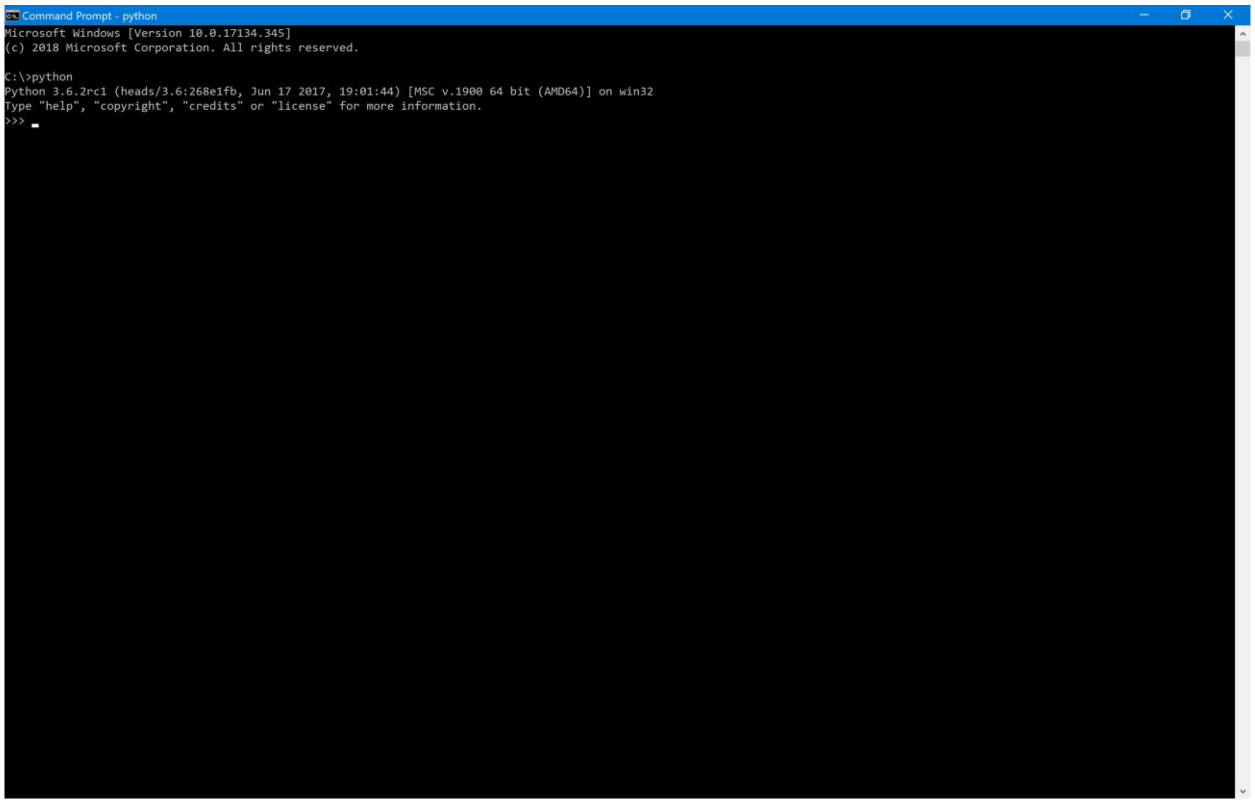
Out [1]



```
Microsoft Windows [Version 10.0.17134.345]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\>
```

A2 – Invoke Python console by writing "python" in the command window

Out [2]



```
Microsoft Windows [Version 10.0.17134.345]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\>python
Python 3.6.2rc1 (heads/3.6:268e1fb, Jun 17 2017, 19:01:44) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

A3 – Import biosignalsnotebooks package

In [3]

```python
import biosignalsnotebooks as bsnb
```

A4 – Generate of the biosignalsnotebooks folder hierarchy (specifying the destination folder)

In [4]

```
bsnb.opensignals_hierarchy("root/dir/of/bsnb")
```

A5 – The previous command will generate the following folders and return the file path of root("root/dir/of/bsnb/biosignalsnotebooks_environment")

Out [5]

```
biosignalsnotebooks_environment root dir
||||> Categories folder that contains subfolders for grouping the Notebooks
|||||||||||||>Detect contains the Notebooks (.ipynb) inside "Detect" category
|||||||||||||>Evaluate contains the Notebooks (.ipynb) inside "Evaluate" category
|||||||||||||>Extract contains the Notebooks (.ipynb) inside "Extract" category
|||||||||||||>Load contains the Notebooks (.ipynb) inside "Load" category
|||||||||||||>MainFiles contains the Notebooks (.ipynb) inside "MainFiles" category
||||||||||||||||||||||||||>aux_files
|||||||||||||>Pre-Process contains the Notebooks (.ipynb) inside "Pre-Process" category
|||||||||||||>Record contains the Notebooks (.ipynb) inside "Record" category
|||||||||||||>Train_and_Classify contains the Notebooks (.ipynb) inside "Train_and_Classify" category
|||||||||||||>Understand contains the Notebooks (.ipynb) inside "Understand" category
|||||||||||||>Visualise contains the Notebooks (.ipynb) inside "Visualise" category
||||>images directory dedicated to store images needed at biosignalsnotebooks environment
|||||||||||||>icons images used for identifying each Notebook category are stored here
||||>signal_samples inside this directory are stored a set of signal samples (.txt and .h5 files)
||||>styles contains CSS files that ensure the correct application of biosignalsnotebooks style
```

A6 – Creation of a "Notebook Object", defining as input arguments the category (*notebook_type*), title (*notebook_title*), list of tags (*tags*), number of stars (*difficulty_stars*) and notebook description (*notebook_description*)

In [6]

```
nb = bsnb.notebook(notebook_type=<str>, notebook_title=<str>, tags=<list>, difficulty_stars=<int>,
        notebook_description=<str>)

# ========================================= Available Options =============================================
# [notebook_type]
# "Load", "Record", "Visualise", "Pre-Process", "Detect", "Extract", "Train_and_Classify", "Understand", "Evaluate" and
# "MainFiles"
#
# [notebook_title]
# All strings are valid inputs
#
# [tags]
# A list where each entry is a different tag. The creator should always include the name of category (lowercase)
# chosen and the acronym defining the type of signal (emg, ecg...) to which the Notebook instruction are
# applicable.
#
# [difficulty_stars]
# 1-5
#
# [notebook_description]
# A string containing a simple Notebook description. For breaking line it should be called the "escape sequence" by
# writing "\n".
```

As a practical example, we will create a Notebook inside "Load" category at 4th difficulty level, which is applicable to EMG signals:

```
nb = bsnb.notebook(notebook_type="Load", notebook_title="A simple template for creating a Notebook",
                   tags=["load", "emg", "test"], difficulty_stars=4, notebook_description="An instructive description,
                                                                                           contextualizing the relevance of the
                                                                                           Notebook")
```

A7 – Storage of the created template inside the biosignalsnotebooks folder hierarchy (created at step 4)

In [7]

```
nb.write_to_file("root/dir/of/bsnb/biosignalsnotebooks_environment", "File_Name")
```

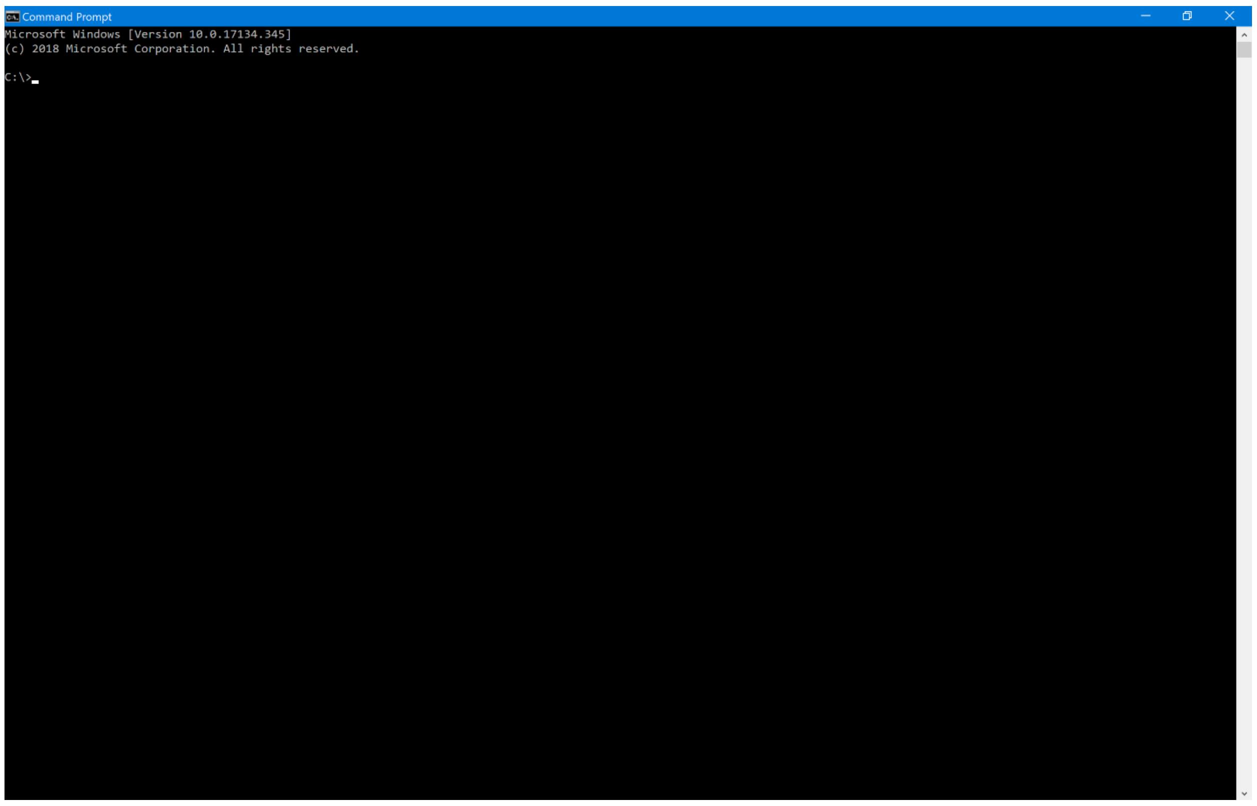
Continuing our practical example:

```
nb.write("root/dir/of/bsnb/biosignalsnotebooks_environment", filename="Load_Test")
```

♂ B - Edit the generated .ipynb file with Jupyter Notebook

B1 – Open a command window (type “cmd” at the start menu)

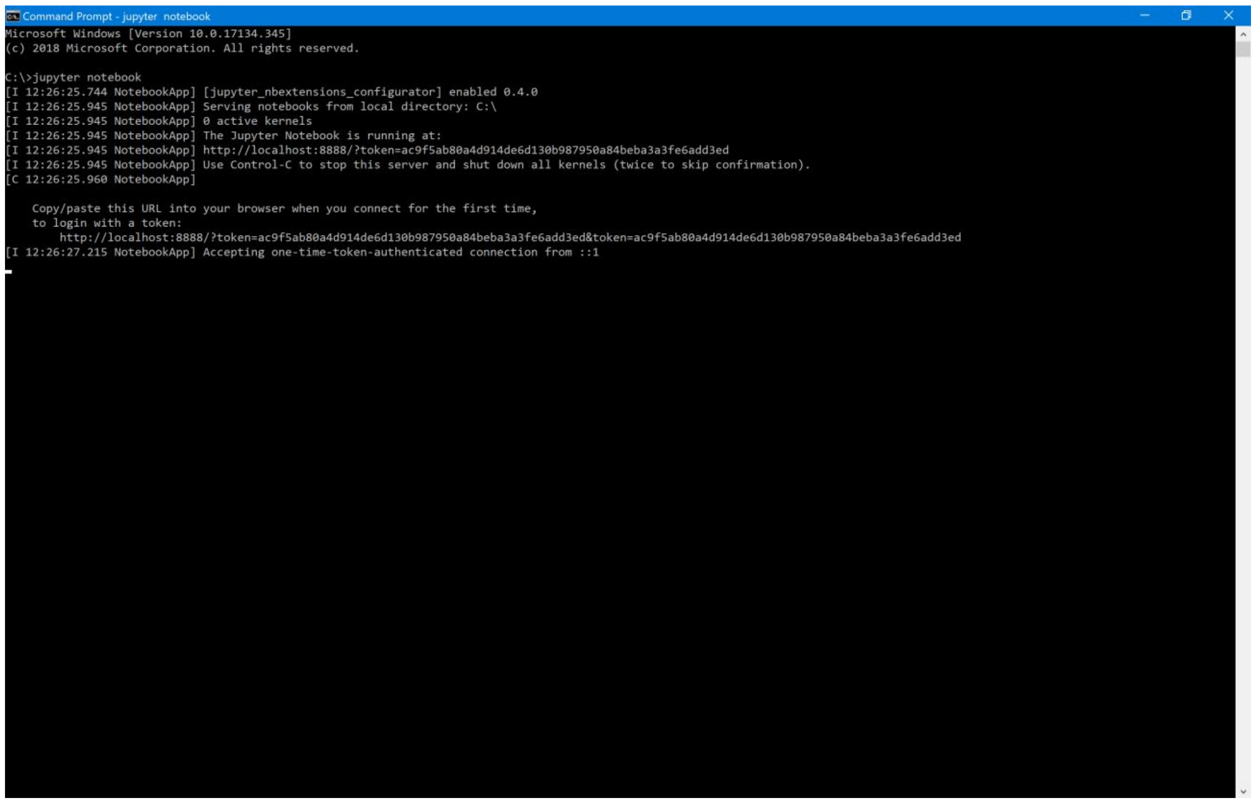
Out [1]



```
Microsoft Windows [Version 10.0.17134.345]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\>_
```

B2 – Invoke Jupyter Notebook by writing “jupyter notebook” in the command window

Out [2]

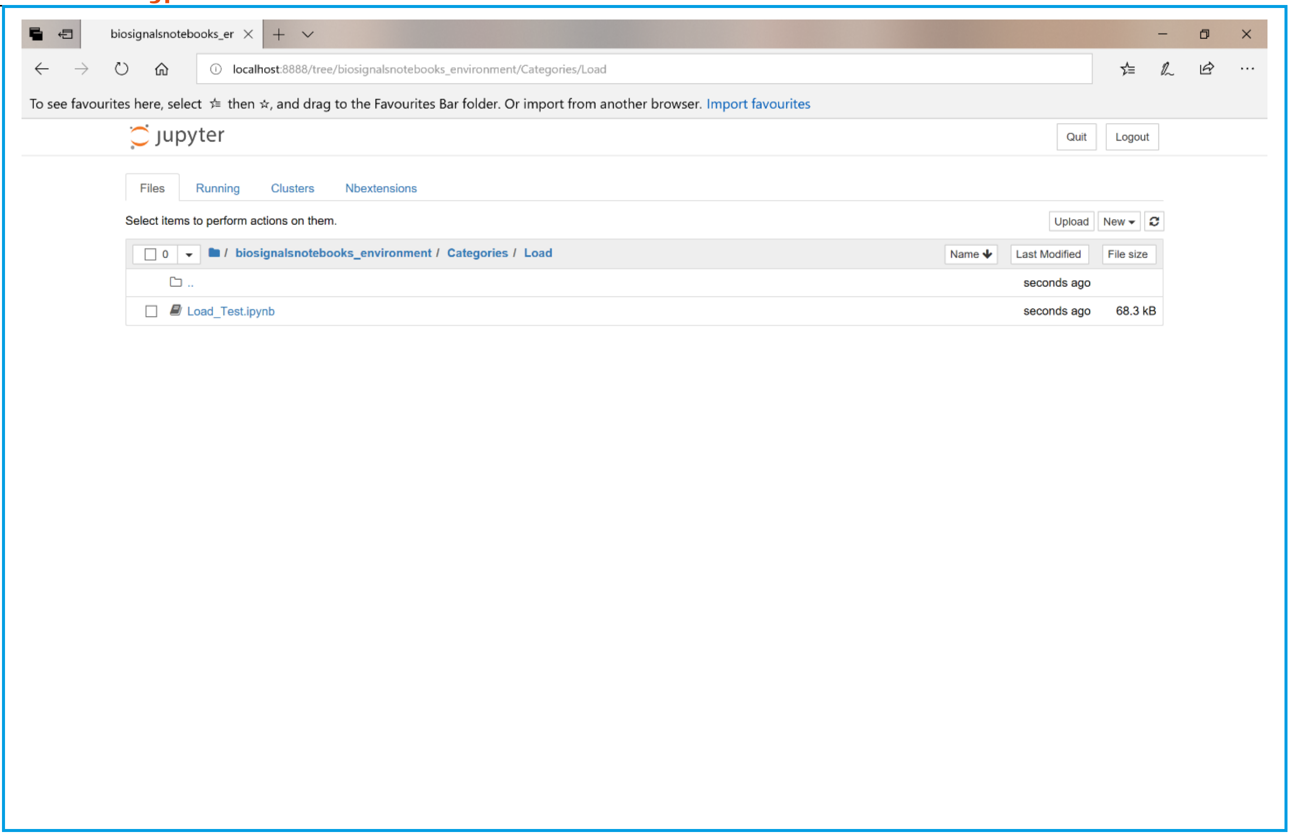


```
Microsoft Windows [Version 10.0.17134.345]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\>jupyter notebook
[I 12:26:25.744 NotebookApp] [jupyter_nbextensions_configurator] enabled 0.4.0
[I 12:26:25.945 NotebookApp] Serving notebooks from local directory: C:\
[I 12:26:25.945 NotebookApp] 0 active kernels
[I 12:26:25.945 NotebookApp] The Jupyter Notebook is running at:
[I 12:26:25.945 NotebookApp] http://localhost:8888/?token=ac9f5ab80a4d914de6d130b987950a84beba3a3fe6add3ed
[I 12:26:25.945 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 12:26:25.960 NotebookApp]

    Copy/paste this URL into your browser when you connect for the first time,
    to login with a token:
        http://localhost:8888/?token=ac9f5ab80a4d914de6d130b987950a84beba3a3fe6add3ed&token=ac9f5ab80a4d914de6d130b987950a84beba3a3fe6add3ed
[I 12:26:27.215 NotebookApp] Accepting one-time-token-authenticated connection from ::1
_
```

A Jupyter Notebook server will be executed locally, and a browser window should arise.

B3 – Navigate through folders until reaching the directory where the generated Notebook file is contained ("root/dir/of/bsnb/biosignalsnotebooks_environment/notebook_type/filename"). As we see in points A6 and A7, notebook_type="Load" and filename="Load_Test".

Out [3]


localhost:8888/tree/biosignalsnotebooks_environment/Categories/Load

To see favourites here, select ☆ then ☆, and drag to the Favourites Bar folder. Or import from another browser. Import favourites

jupyter — Quit — Logout

Files | Running | Clusters | Nbextensions

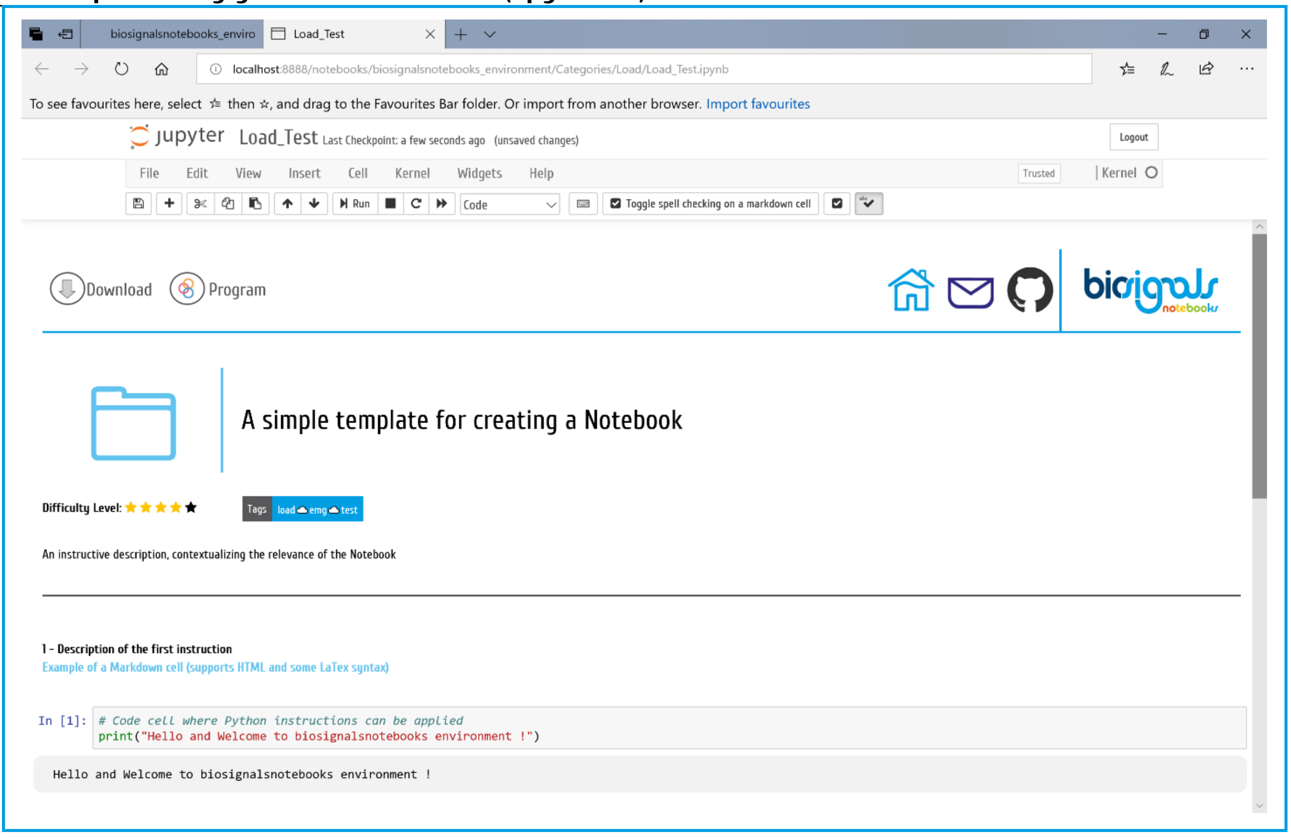
Select items to perform actions on them.

/ biosignalsnotebooks_environment / Categories / Load

Name	Last Modified	File size
..	seconds ago	
Load_Test.ipynb	seconds ago	68.3 kB

B4 – Open the previously generated Notebook (.ipynb file)

Out [4]


localhost:8888/notebooks/biosignalsnotebooks_environment/Categories/Load/Load_Test.ipynb

To see favourites here, select ☆ then ☆, and drag to the Favourites Bar folder. Or import from another browser. Import favourites

jupyter Load_Test Last Checkpoint: a few seconds ago (unsaved changes)

A simple template for creating a Notebook

Difficulty Level: ★★★★★

Tags load ☁ emg ☁ test

An instructive description, contextualizing the relevance of the Notebook

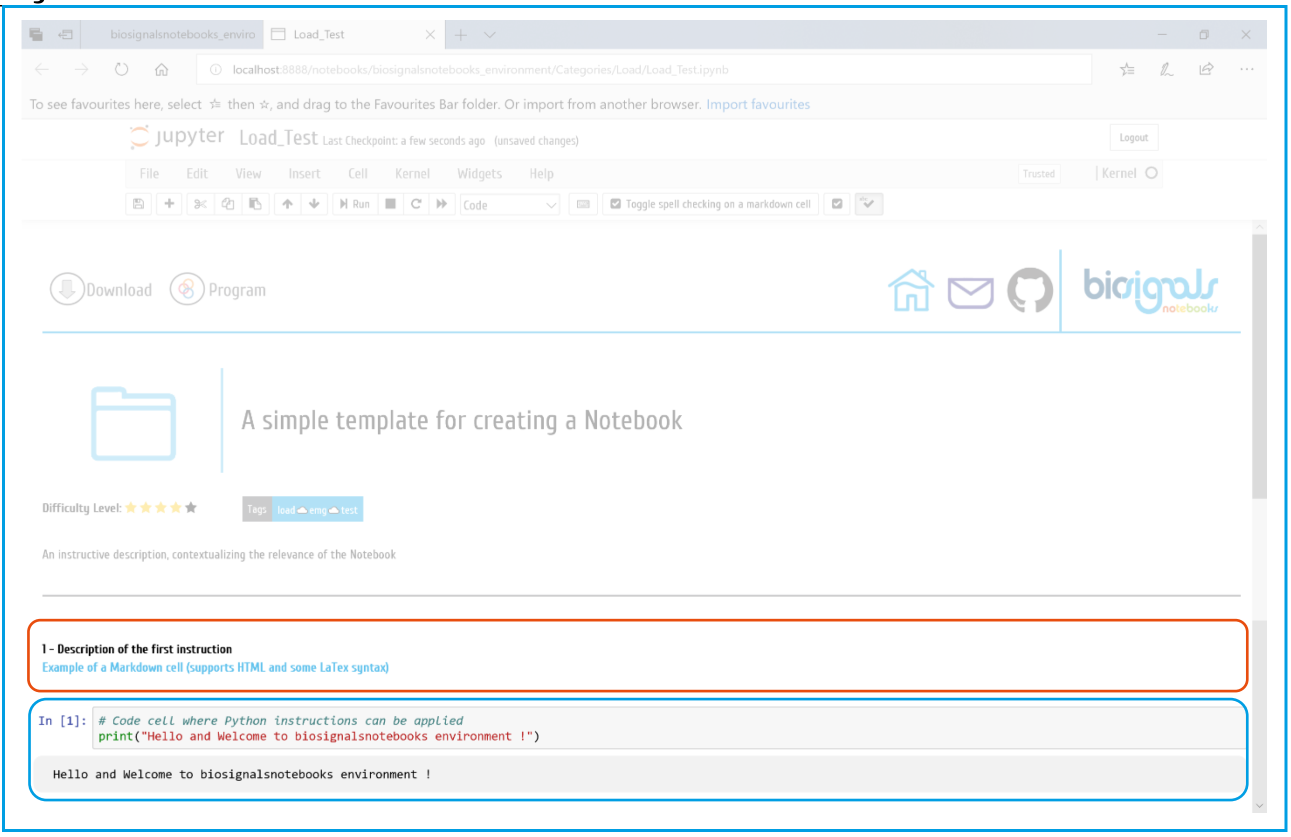
1 - Description of the first instruction

Example of a Markdown cell (supports HTML and some LaTex syntax)

```
In [1]: # Code cell where Python instructions can be applied
        print("Hello and Welcome to biosignalsnotebooks environment !")

Hello and Welcome to biosignalsnotebooks environment !
```

B5 – Now you can fill the Notebook with instructive and attractive contents !

Out [5]


biosignalsnotebooks_enviro
Load_Test
localhost:8888/notebooks/biosignalsnotebooks_environment/Categories/Load/Load_Test.ipynb
To see favourites here, select ☆ then ☆, and drag to the Favourites Bar folder. Or import from another browser. Import favourites
jupyter Load_Test Last Checkpoint: a few seconds ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Kernel
Run
Code
Toggle spell checking on a markdown cell
Download
Program
biosignals notebooks

A simple template for creating a Notebook

Difficulty Level: ★★★★★ Tags load ☁ emg ☁ test

An instructive description, contextualizing the relevance of the Notebook

1 - Description of the first instruction
Example of a Markdown cell (supports HTML and some LaTex syntax)

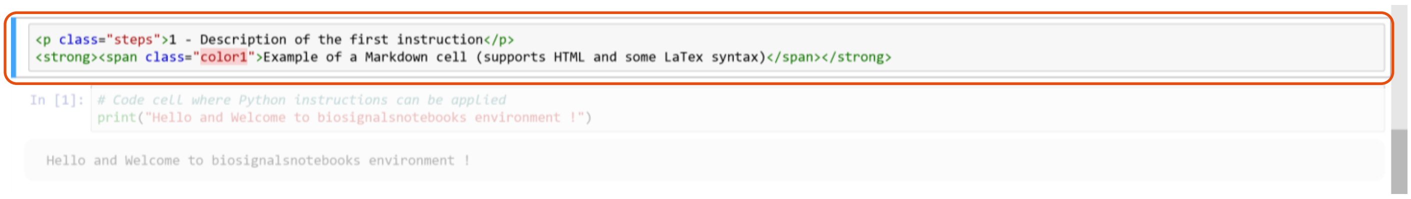
```
In [1]: # Code cell where Python instructions can be applied
        print("Hello and Welcome to biosignalsnotebooks environment !")

   Hello and Welcome to biosignalsnotebooks environment !
```

*Inside **Jupyter Notebook** environment, four types of cells can be created, however, for doing a Notebook with biosignalsnotebooks specifications we only need two of them.*

For specification of text and descriptions it is necessary to create a "Markdown Cell" (the cell highlighted in Red is an example of this type of cell). A "Markdown Cell" supports plain text, markdown, HTML language and some syntax of LaTex.

The second type of cell is a "Code Cell" (highlighted in blue), supporting all Python instructions that you imagine.

We can check the Red cell content by double left-click:



```
<p class="steps">1 - Description of the first instruction</p>
<strong><span class="color1">Example of a Markdown cell (supports HTML and some LaTex syntax)</span></strong>

In [1]: # Code cell where Python instructions can be applied
        print("Hello and Welcome to biosignalsnotebooks environment !")

   Hello and Welcome to biosignalsnotebooks environment !
```

*As can be seen we define a paragraph with <p> tag. It is always necessary to include class="steps" as an argument. This class is defined inside a CSS file, which ensures, for example, that the text will appear in **bold**.*

The second line print a text segment in blue, due to the specification of class="color1" as attribute. There are available the following colours:

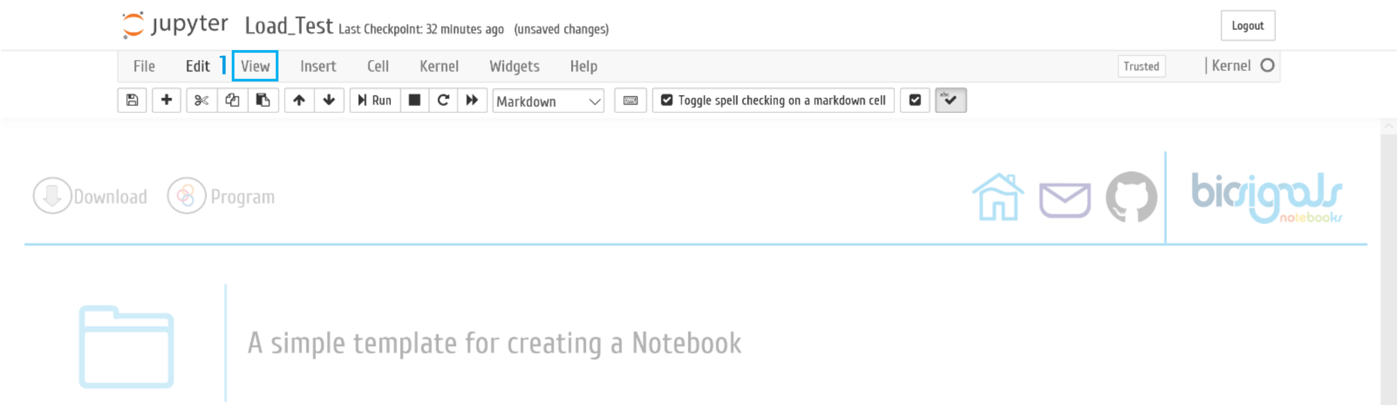
- *class="color1" → "Example of Text"*
- *class="color2" → "Example of Text"*
- *class="color3" → "Example of Text"*
- *class="color4" → "Example of Text"*
- *class="color5" → "Example of Text"*
- *class="color6" → "Example of Text"*
- *class="color7" → "Example of Text"*
- *class="color8" → "Example of Text"*
- *class="color9" → "Example of Text"*
- *class="color10" → "Example of Text"*
- *class="color11" → "Example of Text"*
- *class="color12" → "Example of Text"*

*To apply the changes made to the cell content it is necessary to press together **Ctrl + Enter**.*

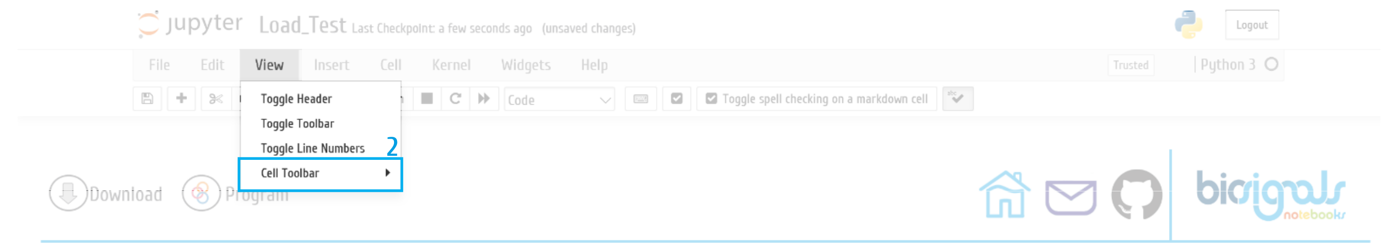
Each cell in formed by an "input" part and an "output" segment, where the results can be shown. But sometimes is important to hide information to the final user.

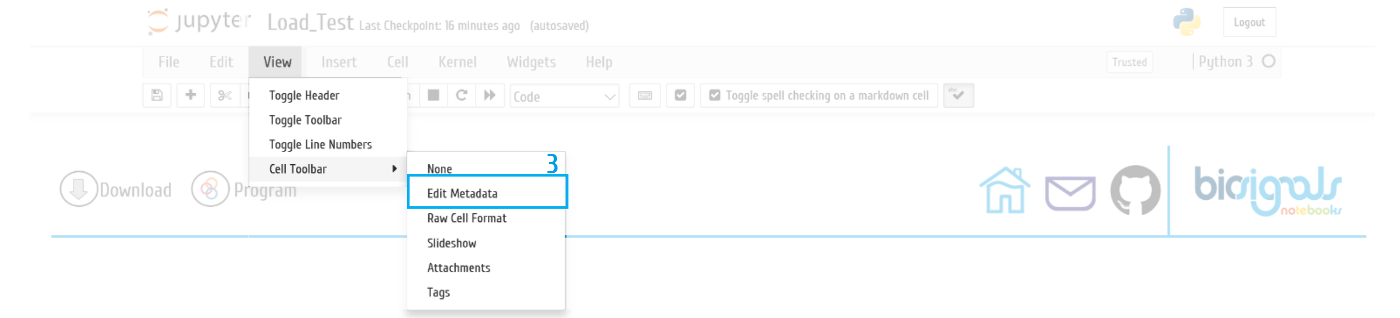
For example, in all Notebooks there are a last cell that contains some JavaScript instructions, responsible for executing all cells automatically, when we load the Notebook.

But this cell should not be visible to the user. To ensure this "invisibility" in the HTML version of the Notebook, we need to access the cell metadata:

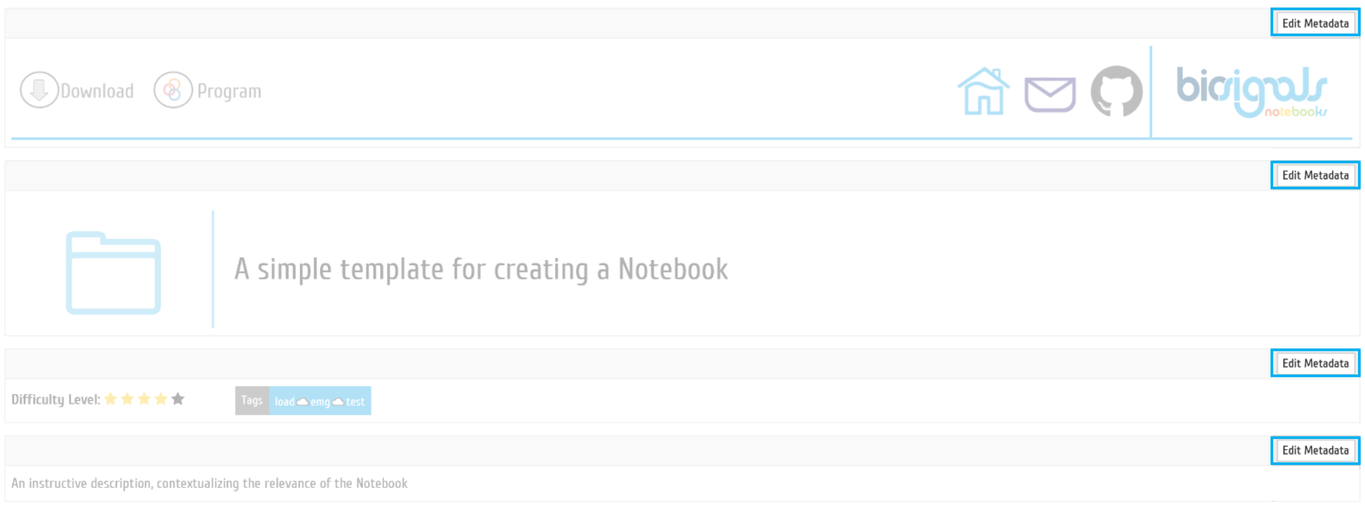

jupyter Load_Test Last Checkpoint: 32 minutes ago (unsaved changes)
Logout
File
Edit
1
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Kernel
Run
Markdown
Toggle spell checking on a markdown cell
Download
Program
biosignals notebooks

A simple template for creating a Notebook





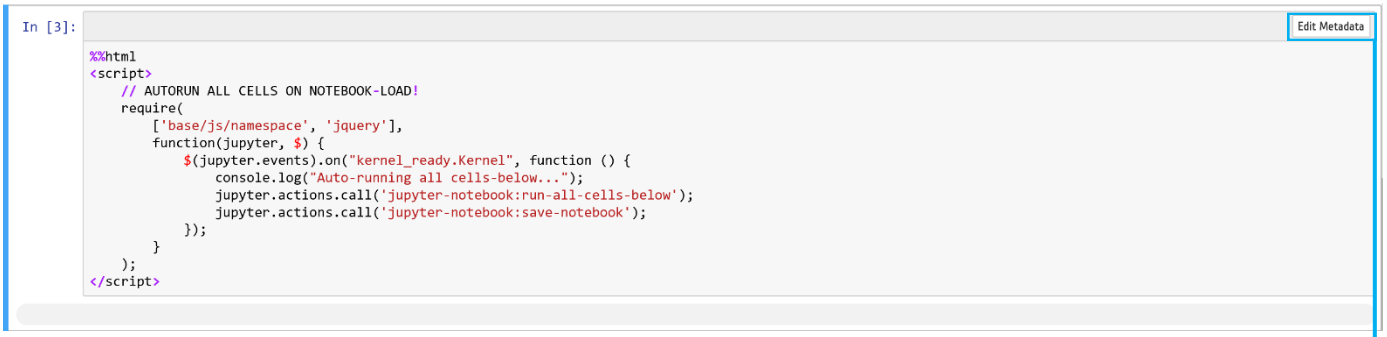
Now, at each cell, a button is available at top right corner for editing metadata of the cell.



A simple template for creating a Notebook

An instructive description, contextualizing the relevance of the Notebook

Let's access the metadata of the last cell of the Notebook

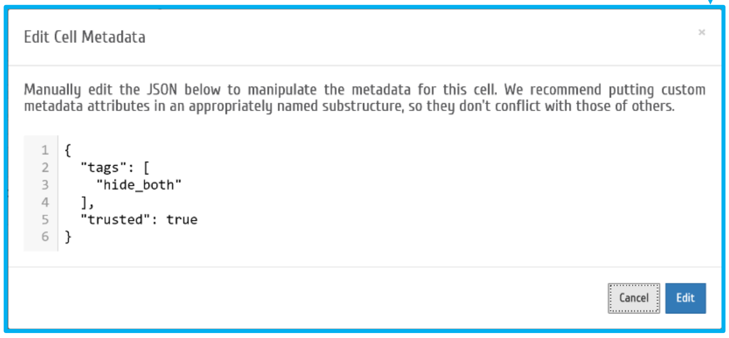


```
%%html
<script>
    // AUTORUN ALL CELLS ON NOTEBOOK-LOAD!
    require(
        ['base/js/namespace', 'jquery'],
        function(jupyter, $) {
            $(jupyter.events).on("kernel_ready.Kernel", function () {
                console.log("Auto-running all cells-below...");
                jupyter.actions.call('jupyter-notebook:run-all-cells-below');
                jupyter.actions.call('jupyter-notebook:save-notebook');
            });
        }
    );
</script>
```

As can be seen, the metadata content is in a json format, with pairs of keys and the respective values.

For now, the "tags" key is the most relevant. Here we can specify one of three values: "hide_both" (for hiding both input and output in HTML version of Notebook), "hide_in" (for hiding the input segment of the cell) and "hide_out" (for hiding the output).

The current cell will be invisible to the user, since both input and output were hidden.



Manually edit the JSON below to manipulate the metadata for this cell. We recommend putting custom metadata attributes in an appropriately named substructure, so they don't conflict with those of others.

```
{
  "tags": [
    "hide_both"
  ],
  "trusted": true
}
```

V1.0. Under continuous development...